Learn Markdown

Published
with GitBook

Table of Contents

Introduction	1.1
About Markdown	1.2
Titles	1.3
Links	1.4
Images	1.5
Code Blocks	1.6
Tables	1.7

Learn Markdown

You may have heard about Markdown, if you have it's a good thing.

Markdown is a plain text formatting syntax designed so that it can optionally be converted to HTML.

In this book, you'll learn how to write document using the markdown syntax.

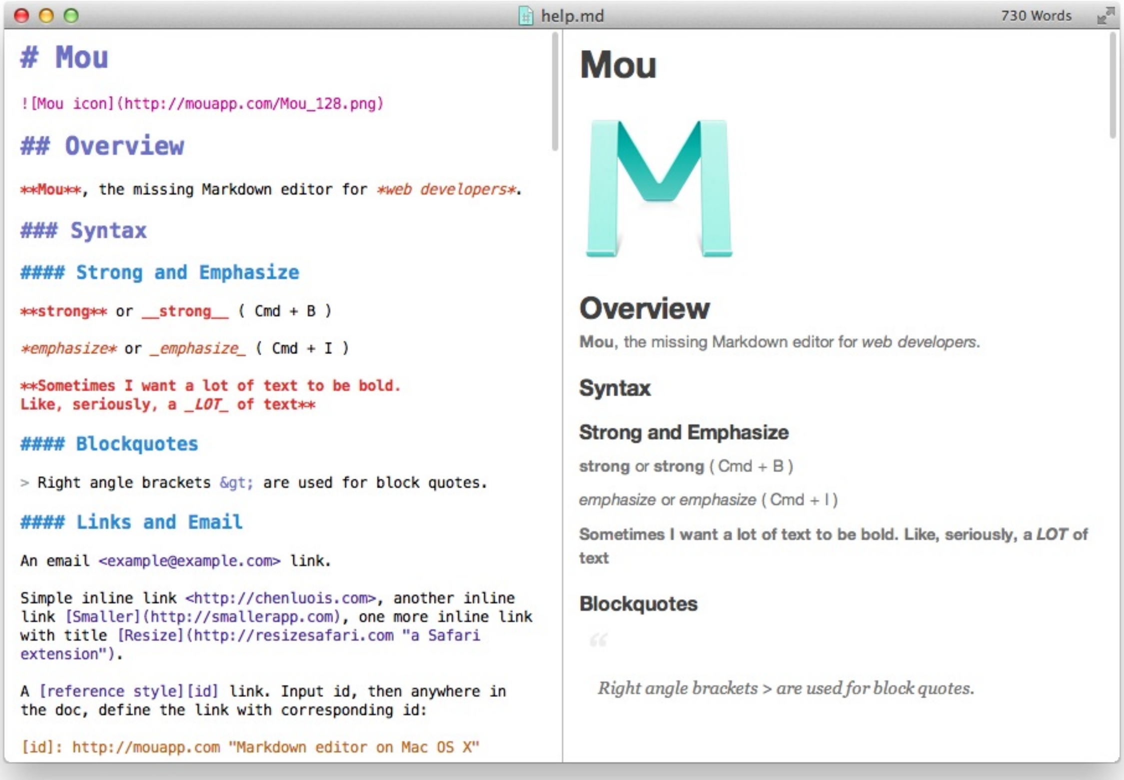

Figure 1: Example of markdown with associated output document on the right.

Short Introduction about Markdown

The Markdown language was created in 2004 by John Gruber with substantial contributions from Aaron Swartz, with the goal of allowing people “to write using an easy-to-read, easy-to-write plain text format, and optionally convert it to structurally valid XHTML (or HTML)”.

Taking cues from existing conventions for marking up plain text in email such as setext, the language was designed to be readable as-is, without looking like it's been marked up with tags or formatting instructions, unlike text which has been formatted with a Markup language, such as HTML, which has obvious tags and formatting instructions. Markdown is a formatting syntax for text that can be read by humans and can be easily converted to HTML.

Gruber wrote a Perl script, Markdown.pl, which converts marked-up text input to valid, well-formed XHTML or HTML and replaces left-pointing angle brackets ('<') and ampersands with their corresponding character entity references. It can be used as a standalone script, as a plugin for Blosxom or Movable Type, or as a text filter for BBEdit.

Markdown has since been re-implemented by others as a Perl module available on CPAN (Text::Markdown), and in a variety of other programming languages. It is distributed under a BSD-style license and is included with, or available as a plugin for, several content-management systems.

Use Cases

Markdown is used in **GitHub**, **GitBook**, **Reddit**, **Diaspora**, **Stack Overflow**, **OpenStreetMap**, and many others.

Even this book is written using Markdown: Raw content of this page.

Files

A markdown document is a text file with the `.md` extension. You can open a markdown file using a simple text editor.

Titles

As we started writing a markdown document, we need to add a title and some sub-headers.

Markdown supports two styles of headers, Setext and atx.

Setext-style headers are “underlined” using equal signs (for first-level headers) and dashes (for second-level headers). For example:

```
This is an H1
=============

This is an H2
-------------
```

Any number of underlining =’s or -’s will work.

Atx-style headers use 1-6 hash characters at the start of the line, corresponding to header levels 1-6. For example:

```
# This is an H1

## This is an H2

###### This is an H6
```

Optionally, you may “close” atx-style headers. This is purely cosmetic — you can use this if you think it looks better. The closing hashes don’t even need to match the number of hashes used to open the header. (The number of opening hashes determines the header level.) :

```
# This is an H1 #

## This is an H2 ##

### This is an H3 ######
```

Here's a quiz about markdown titles.

Select the valid headers:

- [x] `# hello`
- [ ] `#hello`

> Headers need space between the hash characters and the text.

Select the valid headers:

- [ ]

  ```
  test
  ########
  ```

- [x]

  ```
  test
  =======
  ```

> Only '=' and '-' are accepted for underlining an header.

Links

Markdown supports two styles of links: inline and reference.

In both styles, the link text is delimited by [square brackets].

To create an inline link, use a set of regular parentheses immediately after the link text’s closing square bracket. Inside the parentheses, put the URL where you want the link to point, along with an optional title for the link, surrounded in quotes. For example:

```
[I'm an inline-style link](https://www.google.com)

[I'm an inline-style link with title](https://www.google.com "Google's Homepage")

[I'm a reference-style link][Arbitrary case-insensitive reference text]

[I'm a relative reference to a repository file](../blob/master/LICENSE)
```

Reference-style links use a second set of square brackets, inside which you place a label of your choosing to identify the link:

```
This is [an example][id] reference-style link.
```

You can optionally use a space to separate the sets of brackets:

```
This is [an example] [id] reference-style link.
```

Then, anywhere in the document, you define your link label like this, on a line by itself:

```
[id]: http://example.com/  "Optional Title Here"
```

GitHub and **GitBook** supports URL autolinking. They will autolink standard URLs, so if you want to link to a URL (instead of setting link text), you can simply enter the URL and it will be turned into a link to that URL.

Here's a quiz about markdown links.

Select the valid links:

- [x] `[a link](http://google.fr)`

- [ ] `(a link)[http://google.fr]`

> The link text is delimited by [square brackets].

What are the correct informations from this link: `[a link](http://google.fr "google")`

- [ ] the link is https://google.fr
- [x] the title of the link is "google"
- [ ] it'll show the text "google"
- [x] it'll show the text "a link"

> Links can have 3 parts: the text, the url and a title.

Images

```
# Inline
![Alternative text](/path/to/img.jpg "Optional title")

# Reference
![Alternative text][id]
[id]: url/to/image  "Optional title"
```

As you may have noticed, images in Markdown are very similar to links. The difference is that:

- the square brackets must be prefixed with an exclamation mark and
- inside they may have some alternative text. A description of the image, which is displayed if the image can't be loaded.

Here's a quiz about markdown images.

Select the valid images:

- [ ] `[Google logo](https://www.google.ru/logo.png)`
- [x] `![](https://www.google.ru/logo.png)`

> Images must be prefixed with an exclamation mark. The alternative text and a title are optional.

What is true about the following line: `![Funny cat](http://cats.ru/funny.png "Share this")`

- [x] if the url is 404, "Funny cat" will be displayed
- [ ] exclamation mark can be omitted in this case
- [ ] if the url is 404, "Share this" will be displayed
- [x] on mouse over the image "Share this" will be displayed

> Similarly to links, images can have 3 parts: the alternative text, the url and a title. An exclamation mark is nesessary.

Code Blocks

Pre-formatted code blocks are used for writing about programming or markup source code. Rather than forming normal paragraphs, the lines of a code block are interpreted literally.

Here is an example:

```
This is a code block
```

To produce a code block in Markdown, simply indent every line of the block by at least 4 spaces or 1 tab.

For example:

```
This is a normal paragraph:

    This is a code block.
```

You can also create code block separated by:

````
```
````

Inline code blocks

Inline code blocks can be written using: `

For example:

```
This is a `inline code block`
```

Syntax highlighting

You can define the language to be used for syntax highlighting by adding the name on the opening tag. Example:

````
```js
var a = {};
```
````
````

Tables

Tables aren't part of the core Markdown spec, but they are part of GFM (GitHub Markdown) and Markdown Here supports them.

Here is an example of table with the output below:

```
| Tables        | Are           | Cool  |
| ------------- |:-------------:| -----:|
| col 3 is      | right-aligned | $1600 |
| col 2 is      | centered      |   $12 |
| zebra stripes | are neat      |    $1 |
```

Tables	Are	Cool
col 3 is	right-aligned	$1600
col 2 is	centered	$12
zebra stripes	are neat	$1

Colons can be used to align columns.

The outer pipes (|) are optional, and you don't need to make the raw Markdown line up prettily. You can also use inline Markdown.

Example:

```
Markdown | Less | Pretty
--- | --- | ---
*Still* | `renders` | **nicely**
1 | 2 | 3
```